A declarative query language for statistical inference

Gitte Vanwinckelen, Hendrik Blockeel

Department of Computer Science, KU Leuven, Belgium

1 Introduction

Large volumes of experimental data are generated each day in computational sciences such as bioinformatics, chemistry, and physics. Statistical analyses are no longer only restricted to statisticians, but are increasingly being performed by non-experts [8]. The process is not only labor-intensive but also error-prone. Statistics provides us with a plethora of data analysis and inference methods, making it practically impossible for a scientist to have full knowledge of the existing methods and statistical assumptions that have to be satisfied to apply them. Scientists are often even unaware when a statistical assumption is violated.

We propose to tackle these problems with a formal declarative language in which experimental questions, and the necessary background information, can be formulated. A scientist could then formulate a hypothesis as a query, after which the remainder of the experimental process is performed automatically. This approach allows a scientist to focus on a correct understanding of high level concepts from statistics rather than technical details.

To clarify our goal, we start by comparing this idea to those underlying existing systems. Next, we propose a preliminary design of the language by looking into one specific task, namely, the computation of a confidence interval for the population mean of a random variable.

2 Related work

A great amount of software exists for machine learning and statistics, e.g., Weka and R. These packages require a thorough understanding of statistics to perform meaningful inferences. For instance, they will not give a warning if a user performs a hypothesis test on a sample that is too small to get useful results.

Statistical expert systems aim to remedy these problems by giving advice on the design and analysis of an experiment [5]. While they have the same goal as our declarative experimentation system, the implementation differs. Statistical expert systems are often based on interaction with natural language yes/no questions, and the control system is typically implemented with IF THEN rules. Our solution, on the contrary, allows the user to formulate a query in a formal declarative language together with the necessary constraints and assumptions. A database is used to perform statistical inference and answer the query.

While our system is database oriented, it is different from a statistical database system. Such a system allows the user to perform statistical analyses on a database by computing statistical aggregates, but it disallows access to individual records. Research in this area focuses mostly on data anonymization [2].

Also related are probabilistic databases, which represent uncertainty in the database [4]. It should be noted that our approach is not concerned with uncertainty in the database itself. We assume a given deterministic database that is seen as a sample from a population, and want to query that population, not the database itself. This crucially sets apart the two approaches.

An idea that is more closely related, is the construction of probabilistic models from a relational database by Singh and Graepel [9]; these models then describe the population that the database is a sample from. Methods such as this one will play a role in any implementation of our approach. We here focus, however, on creating a language and execution mechanism that supports a general type of queries, thus offering flexibility and ease of use to the user.

Lastly, our system is related to inductive database systems, which aim to integrate data mining and machine learning into database management systems [6]. They allow for users to query a database for patterns by formulating questions in a declarative language rather than running a predefined algorithm. An inductive database system for constraint based clustering was recently proposed by Adam and Blockeel [1]. While inductive database systems focus on inference with machine learning models, our focus is on statistical inference.

3 Query language design

The language requirements can be looked at from two viewpoints. From the statistical viewpoint, we want to formulate queries in a declarative manner to shield the user from low level choices, e.g., the choice of the method to compute a confidence interval for a population parameter. From the database viewpoint, our query language is based on SQL, but we want to query statistical populations, instead of a finite database. We illustrate the language design by introducing and explaining the 'ESTIMATE' statement.

$$
\begin{aligned}
\langle \mathit{ExpQL} \rangle ::= \ & \text{ESTIMATE } \langle \mathit{population\_statistic} \rangle \\
& \text{FROM } \langle \mathit{sample} \rangle \\
& [\ \text{WHERE } \langle \mathit{condition} \rangle\ ] \\
& [\ \text{ENSURING } \langle \mathit{statistical\_accuracy} \rangle\ ]
\end{aligned}
$$

$$
\langle \mathit{population\_statistic} \rangle ::= \text{PROP } \langle \mathit{data} \rangle \mid \text{MEAN } \langle \mathit{data} \rangle
$$

$$
\langle \mathit{statistical\_accuracy} \rangle ::= \text{CONF } \langle \mathit{confidence} \rangle
$$

The population parameter that we want to estimate is given by $\langle \mathit{population\_statistic} \rangle$. $\langle \mathit{sample} \rangle$ is the data that we have available to estimate the population parameter. It is a database table that has an attribute-value structure. With the WHERE $\langle \mathit{condition} \rangle$ clause we can specify a subpopulation. The ENSURING $\langle \mathit{statistical\_accuracy} \rangle$ clause imposes constraints on the population parameter estimator. For now, we focus on confidence intervals, and the constraints can apply to the confidence level, the width, or a combination of both.

4 Query execution

We illustrate the query execution with an example. Consider a database table *employee* that contains employees of a multinational corporation. Each record

consists of the following properties: Id, name, length, gender, nationality, and hair color. We are interested in a 95% confidence interval with a maximum width of 5cm for the expected length of all Swedish male employees with blond hair:

```
ESTIMATE MEAN length
FROM employee
WHERE gender= 'male' AND nationality='Swedish' AND haircolor='blond'
ENSURING CONF=0.95 AND WIDTH <= 5;
```

If a user specifies both a confidence level and a maximum width, this puts a constraint on the minimum size of the sample. If the sample is too small, we propose two alternative execution strategies to still answer the query.

First, it may be possible to couple the query system to a data generator, and use active learning to generate the necessary data. The data generator may be an actual physical experimentation system. An existing example is the Robot Scientist, which combines physical execution with active learning and automated hypothesis generation [7]. If this is not possible, the system can notify the user about the shortage of data, and request him or her to collect more data. It can still assist the user with the data collection, for instance, by computing the minimum number of samples needed.

Second, we can try to incorporate more data in an intelligent manner by relaxing the constraints. In our example query, the user assumes length is dependent on hair color, gender, and nationality, but perhaps one or more of these dependencies is very weak. For instance, some relationship exists between hair color and length; Scandinavians are on average taller than South Europeans, and are also blond more often. However, it is unclear if this relationship still holds for Swedish men only. If we know hair color and length are independent for Swedish men, we can remove the condition for blond hair and take into account men of any hair color to compute the confidence interval for the mean length.

To apply this approach, we need a method to detect independence between two variables. Many different tests are known for this purpose: The Chi-square test, the Student t-test, etc. Which should be used depends on the types of variables, i.e, categorical or continuous, and their distributions.

In our preliminary experiments on data collected from the UCI repository [3], we investigate constructing a confidence interval for the mean of a continuous variable, which is dependent on a binary variable. A first approach is to test for independence with a t-test that tests for the difference between the means of two independent samples. However, one should be careful with null hypothesis testing. If the null hypothesis is not rejected we have not proven it, instead have insufficient evidence to disprove it. An alternative procedure, which we are still looking into, that does not suffer from this problem is equivalence testing [10].

There is a probability that the test will not detect dependence. This will cause a discrepancy between the imposed confidence level from the query, and the true confidence level. To still provide the user with a correct answer, we aim to quantify this difference. Our experiments are a first step towards this goal.

The experiments indicate that when a t-test does not detect a difference between the two means for different values of the binary variable, we can safely

include the extra data to compute the confidence interval. Because of the additional data, the interval width decreases and this helps us provide a confidence interval with both the required confidence level and maximum width. When the t-test does detect a difference between the means, however, the confidence level is significantly smaller than the requested level, so the approach is not applicable.

5 Conclusion

We presented preliminary ideas on the design of an experimentation system that consists of a declarative language and an inference engine. The language would allow to formulate a hypothesis about a data population, whereafter the inference engine automatically provides an answer, based on a limited sample. In a first stage of this research we introduced the ESTIMATE statement that can be used to formulate a query about a population statistic. We focused on the computation of a confidence interval for the population mean. The system is responsible for choosing the appropriate execution strategy to satisfy the requested confidence level and width. To this purpose, we proposed two different approaches; supplementing the database with new data, or attempting to relax some of the constraints that the user imposed on the data.

We plan to extend this work with additional queries in order to arrive at a complete language. Our focus lies on queries relevant to machine learning researchers. Furthermore, while the preliminary language design is based on SQL, we also plan to investigate probabilistic logic programming languages (e.g., Problog), and first order knowledge representation languages (e.g., FO(.)).

References

1. A. Adam and H. Blockeel. A query language for constraint-based clustering. *Benelearn*, pages 1–7, 2013.
2. N.R. Adam and J.C. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Computing Surveys*, 21(4):515–556, 1989.
3. K. Bache and M. Lichman. UCI repository. http://archive.ics.uci.edu/ml, 2013.
4. N. Dalvi, C. Ré, and D. Suciu. Probabilistic databases: diamonds in the dirt. *Communications of the ACM*, 52(7):86–94, July 2009.
5. D.J. Hand. Expert systems in statistics. *The Knowledge Engineering Review*, 1:2–10, 1984.
6. T. Imielinski and H. Mannila. A database perspective on knowledge discovery. *Communications of the ACM*, 39(11):58–64, 1996.
7. R. King, M. Young, A. Clare, K. Whelan, and J.J. Rowland. The robot scientist project. In *Discovery Science*, pages 16–25, 2005.
8. J. Leek. The vast majority of statistical analysis is not performed by statisticians. http://simplystatistics.org/2013/06/14/the-vast-majority-of-statistical-analysis-is-not-performed-by-statisticians/, 2013. Accessed: 2013-06-24.
9. S. Singh and T. Graepel. Compiling relational database schemata into probabilistic graphical models. In *Neural Information Processing Systems (NIPS), Workshop on Probabilistic Programming*, 2012.
10. D. L. Streiner. Unicorns do exist: a tutorial on "proving" the null hypothesis. In *A guide to the statistically perplexed:selected readings for clinical researchers*, pages 211–223. University of Toronto Press, 2013.